



Jest

Quick start

```
npm install --save-dev jest babel-jest
```

```
/* Add to package.json */
"scripts": {
  "test": "jest"
}
```

```
# Run your tests
npm test -- --watch
```

See: [Getting started](#)

Writing tests

```
describe('My work', () => {
  test('works', () => {
    expect(2).toEqual(2)
  })
})
```

See: [describe\(\)](#), [test\(\)](#), [expect\(\)](#)

BDD syntax

```
describe('My work', () => {  
  it('works', () => {  
    ...  
  })  
})
```

`it` is an alias for `test`. See: [test\(\)](#).

Setup

```
beforeEach(() => { ... })  
afterEach(() => { ... })
```

```
beforeAll(() => { ... })  
afterAll(() => { ... })
```

See: [afterAll\(\)](#) and more

Focusing tests

```
describe.only(...)  
it.only(...) // alias: fit()
```

See: [test.only](#)

Skipping tests

```
describe.skip(...)  
it.skip(...) // alias: xit()
```

See: [test.skip](#)

Optional flags `--coverage` See a summary of test coverage `--detectOpenHandles` See a summary of ports that didn't close `--runInBand` Run all tests one after the other

#Expect

Basic expectations

```
expect(value)  
  .not  
  .toBe(value)  
  .toEqual(value)  
  .toBeTruthy()
```

Note that `toEqual` is a deep equality check. See: [expect\(\)](#)

Snapshots

```
expect(value)
  .toMatchSnapshot()
  .toMatchInlineSnapshot()
```

Note that `toMatchInlineSnapshot()` requires Prettier to be set up for the project. See: [Inline snapshots](#)

Errors

```
expect(value)
  .toThrow(error)
  .toThrowErrorMatchingSnapshot()
```

Booleans

```
expect(value)
  .toBeFalsy()
  .toBeNull()
  .toBeTruthy()
  .toBeUndefined()
  .toBeDefined()
```

Numbers

```
expect(value)
  .toBeCloseTo(number, numDigits)
  .toBeGreaterThan(number)
  .toBeGreaterThanOrEqual(number)
  .toBeLessThan(number)
  .toBeLessThanOrEqual(number)
```

Objects

```
expect(value)
  .toBeInstanceOf(Class)
  .toMatchObject(object)
  .toHaveProperty(keyPath, value)
```

Objects

```
expect(value)
  .toContain(item)
  .toContainEqual(item)
  .toHaveLength(number)
```

Strings

```
expect(value)
  .toMatch(regexOrString)
```

Others

```
expect.extend(matchers)
expect.any(constructor)
expect.addSnapshotSerializer(serializer)

expect.assertions(1)
```

#More features

Asynchronous tests

```
test('works with promises', () => {
  return new Promise((resolve, reject) => {
    ...
  })
})

test('works with async/await', async () => {
  const hello = await foo()
  ...
})
```

Return promises, or use async/await. See: [Async tutorial](#)

Snapshots

```
it('works', () => {
  const output = something()
  expect(output).toMatchSnapshot()
})
```

First run creates a snapshot. Subsequent runs match the saved snapshot.

See: [Snapshot testing](#)

React test renderer

```
import renderer from 'react-test-renderer'

it('works', () => {
```

```
const tree = renderer.create(
  <Link page="http://www.facebook.com">Facebook</Link>).toJSON()

expect(tree).toMatchSnapshot()
})
```

React's test renderer can be used for Jest snapshots. See: [Snapshot test](#)

Timers

```
jest.useFakeTimers()

it('works', () => {
  jest.runOnlyPendingTimers()
  jest.runTimersToTime(1000)
  jest.runAllTimers()
})
```

See: [Timer Mocks](#)

#Mock functions

Mock functions

```
const fn = jest.fn()
const fn = jest.fn(n => n * n)
```

See: [Mock functions](#)

Assertions

```
expect(fn)
  .toHaveBeenCalled()
  .toHaveBeenCalledTimes(number)
  .toHaveBeenCalledWith(arg1, arg2, ...)
  .toHaveBeenLastCalledWith(arg1, arg2, ...)
```

```
expect(fn)
  .toHaveBeenCalledWith(expect.anything())
  .toHaveBeenCalledWith(expect.any(constructor))
  .toHaveBeenCalledWith(expect.arrayContaining([ values ]))
  .toHaveBeenCalledWith(expect.objectContaining({ props }))
  .toHaveBeenCalledWith(expect.stringContaining(string))
  .toHaveBeenCalledWith(expect.stringMatching(regex))
```

Instances

```
const Fn = jest.fn()
```

```
a = new Fn()  
b = new Fn()  
Fn.mock.instances  
// → [a, b]
```

See: [.mock_property](#).

Calls

```
const fn = jest.fn()  
fn(123)  
fn(456)  
fn.mock.calls.length // → 2  
fn.mock.calls[0][0] // → 123  
fn.mock.calls[1][0] // → 456
```

See: [.mock_property](#).

Return values

```
const fn = jest.fn(() => 'hello')  
  
or:  
jest.fn().mockReturnValue('hello')  
jest.fn().mockReturnValueOnce('hello')
```

Mock implementations

```
const fn = jest.fn()  
  .mockImplementationOnce(() => 1)  
  .mockImplementationOnce(() => 2)  
fn() // → 1  
fn() // → 2
```